REMARKS

Applicants amend claims 1, 5, 12, 17, 24, 32, 34, 36, 43, 48, 52, 57, 66, 70 and 74. Claims 7-11 are canceled herein, and claim 19 has been previously canceled. Hence, claims 1-16, 12-18 and 20-78 are pending, of which claims 1, 5, 12, 17, 24, 32, 34, 43, 52, 61, 64, 66, 70 and 74 are independent. No new matter is added. Applicants respectfully submit that the pending claims define over the art of record.

Interview Summary

Applicants thank the Examiner for the opportunity to meet and discuss the present application. Present at the interview conducted on March 6th, 2007, were Vijaya Raghavan – a co-inventor, Kevin Canning, Reg. No. 35,470 – attorney of record for the applicants, and Lucy Lubashev, Reg. No. 55,408 -- attorney for The MathWorks, Inc.

During the interview, Applicants discussed the history of finite state machine and state charts, and the state of the art prior to the Applicants' invention. Applicants have also explained the significance of the claimed invention and its difference from the cited art. The Examiner has contended that some of the features of the invention need to be clarified or more particularly pointed out in the claims. The Examiner has stated that Figure 19 of Kodosky (United States Patent Publication No. 2002/0083413 to Kodosky et al.) shows some or all of the claimed limitations. Applicants have respectfully disagreed.

In a brief overview, Vijaya Raghavan talked about finite state machines and their properties: a finite state machine is a computational model that may be used to express a broad range of computational problems. A finite state machine typically consists of one or more states and one or more transitions between those states. A "state chart" is a finite state machine represented in such a way that it may be executed using a computing device in order to simulate the behavior of the system it models. A state chart may be hierarchical – that is, containing multiple levels of state machine elements, and/or transitions between states may contain one or more decision junction, such that a transition from one state may lead back to the same state or one of one or multiple other states, depending on conditions on the transition and/or its junctions. One or more actions may be associated with the state or transition; those actions may be

executed when the corresponding state is active or when the corresponding transition is taken, based on the rules for executing the state and transition actions.

Vijaya Raghavan explained that, prior to Applicants' invention, all the actions available for execution at a particular state or transition had to be specifically associated with that state. There existed no mechanism for associating a repeatable group of actions with more than one state and/or transition in a model without replicating that group of actions at each such state and/or transition. In the course of reducing their invention to practice, Applicants have realized that it would be useful to have an ability to associate a group of actions with one or more points in the state chart without having to replicate those actions in every place they are needed. To that end, Applicants have come up with a concept of abstracting one or more actions into a function that may be called from anywhere within the model. Moreover, the function that is called is defined graphically – that is, through a number of graphical elements, possibly in combination with some textual elements. Each graphical function is assigned a name, which may be used to call that graphical function. Using graphical functions, instead of textually-defined functions, may be beneficial for a number of reasons, such as, for example: using the same graphical environment to define the function as the model itself, users familiarity with graphical environments as opposed to textual programming, ease of conveying some functional concepts through graphical means, etc. While the graphical function is defined graphically, it may be called textually from any number of places within an executable model.

During the interview, Vijaya Raghavan has demonstrated a few graphical functions, their definition and use within the model, to Examiners Alhija and Shah. Examiner Alhija said that he believes that Figure 19 of Kodosky shows a graphical function used within the state chart. Applicant and his representatives disagreed and argued that: Figure 19 of Kodosky illustrates the whole of a state chart, not a group of actions that may be associated with one state or transition inside the chart. Furthermore, even if Kodosky could be seen as showing a group of actions associated with a state, as proposed by the Examiner, Kodosky still would fail to anticipate or render obvious claims of the present application, because nowhere does Kodosky show that this group is a function and/or that it may be called from a state or transition. At most, Kodosky shows that a number of actions may be associated with a state. As explained by Vijaya

Raghavan, being able to graphically define a function and call that function from one or more places within a model is distinct from simply associating one or more actions with a state.

At the conclusion of the interview, it was agreed on that Applicants would review and potentially revise the claims to address some of the points discussed during the interview. Accordingly, Applicants amend a number of the pending claims in the present amendment in order to further clarify and point out what they regard as the invention.

Claim Rejections

Claims 1-18 and 20-78 stand rejected under 35 U.S.C. §102(e) as being anticipated by United States Patent Publication No. 2002/0083413 to Kodosky et al. (hereafter "Kodosky"). Applicants respectfully submit that Kodosky does not disclose each and every element of the pending claims.

The Claimed Invention

The claimed invention allows a user to use a diagram to visually represent a function that may be called from a graphical model. The graphical model may be a state chart. An advantage of the claimed invention is that the diagrammatic representation of the function can be easier to understand and modify than a textual representation. Furthermore graphical functions can be called from multiple places within the model. For example, in case of a model represented as a state chart, one or more graphical functions may be called from one or more states and/or transitions in the model. While graphical functions are defined graphically, they may be invoked textually. Graphical functions allow for reuse without replication within a graphical model.

Kodosky

Kodosky disclosed a system for programmatically generating a graphical program in response to state diagram information. The state diagram information may specify a plurality of states and state transitions, which are then converted to a graphical program using a graphical program generation program (GPG program). (Abstract, ll. 1-9). The automatically generated graphical program may be further modified by a user with source code that specifies execution

instructions for each state and Boolean conditions for each state transition. (Abstract, ll. 14-19). That is, Kodosky merely teaches that a non-executable representation of a finite state machine model may be converted to an executable graphical program representing the same model. In addition, users of the system of Kodosky may explicitly associate actions or conditions with states and transitions, respectively. Nowhere does Kodosky teach or suggest that a graphical executable function may be defined such that it may be called from one or more places in the model.

Claims 1-4 and 6

Kodosky does not teach or suggest the limitations of "defining at least one function to be used in a graphical representation of a finite state machine" or "calling the function that is represented graphically from at least two places within the graphical representation of the finite state machine."

The Examiner states that Kodosky discloses the step of defining at least one function within a graphical representation of a finite state machine at page 14, paragraph 165, lines 1-5 and figure 19. Applicants respectfully disagree. This section merely discloses an exemplary graphical program that is programmatically generated based on a state diagram. Were this generated graphical program to be considered a "function," *arguendo*, as per Examiner's interpretation, then Kodosky would fail to show the finite state machine itself. That is, inherent in the claim limitation of "defining at least one function to be used in a graphical representation of a finite state machine" is the requirement that there be present at least two distinct entities: (1) the "finite state machine" and (2)"at least one graphical function." The finite state machine illustrated in Figure 19 of Kodosky cannot be held as anticipatory for both of those two distinct entities, much less for the ability to call one entity from another.

Therefore, Kodosky at best discloses the equivalent of "a graphical representation of a finite state machine," but does not disclose the step of defining at least one graphical function.

Furthermore, Kodosky does not teach or suggest "calling the function that is represented graphically from at least two places within the graphical representation of the finite state machine." As discussed above, Kodosky does not disclose a graphical function at all. At best,

Kodosky teaches that a state may have a number of executable instructions associated with it. However, a function is different from a mere listing of one or more instructions. For example, The Authoritative Dictionary of IEEE Standards and Terms, Seventh Edition, defines a function as follows:

"Function: (1)(B) (software): a software module that performs a specific action, is invoked by the appearance of its name in an expression, may receive input values, and returns a single value."

Nothing in Kodosky indicates that any of the graphical elements may compose a module that is invoked by the appearance of its name in an expression. In contrast, claim 1 specifically recites "calling the function that is represented graphically." In addition, amended claim 1 includes the limitation of "calling the function... from at least two places within the graphical representation of the finite state machine." Kodosky does not teach or suggest anything comparable, nor, indeed, can it, due to the lack of the teachings of the graphical function itself.

Accordingly, Kodosky does not disclose each and every element of independent claim 1. Applicants respectfully request that the Examiner reconsider and withdraw the rejection of claim 1 and its dependent claims 2-4 and 6.

Claim 5

Now independent claim 5 requires the limitation of "defining at least one function to be used in a graphical representation of a finite state machine," where "the function is represented graphically as a diagram comprising graphical elements."

As discussed above, Kodosky does not teach or suggest defining at least one graphical function. Furthermore, nowhere does Kodosky show that such function can be represented graphically as a diagram comprising graphical elements. The Examiner suggests that Kodosky discloses the function that is represented graphically as a diagram comprising graphical elements in Figure 8. However, shown in Figure 8 is the graphical program source code that is generated based on the initial (non-executable) state diagram. (Kodosky, ¶0048). That is, Kodosky itself states that Figure 8 is a representation of the entire state diagram, not a function that may be used in a graphical representation of that state diagram, as required by Claim 5.

Accordingly, Kodosky does not disclose each and every element of independent claim 5. Applicants respectfully request that the Examiner reconsider and withdraw the rejection of claim 5.

Claims 12-16

Independent claim 12 requires the element of a computer program residing on computer readable media having instructions to cause the computer to "use the at least one graphical function in a simulation of a system represented by the finite state machine, wherein the instructions to use the at least one graphical function further comprise instructions to call the at least one graphical function from at least one state or transition in the finite state machine."

The Examiner asserts that Kodosky discloses the limitation of "use at least one graphical function in a simulation of a system represented by the finite state machine" in Kodosky at page 1, paragraph 9, lines 1-2. Applicants are puzzled by that assertion. The cited section states that: "The method disclosed in Kodosky et al. [U.S. Pat. No. 4,901,221] allows a user to construct a diagram using a block diagram editor." Applicants find no discernible correspondence between allowing a user to construct a diagram using a block diagram editor, as admittedly known in the prior art, and using at least one graphical function in a simulation of a system. Applicants respectfully request that the Examiner reconsider and withdraw the rejection of claim 12. Dependent claims 13-16 depend on independent claim 12 and are not anticipated by Kodosky for at least the same reasons as above. Applicants respectfully request withdrawal of the rejection of claims 13-16.

Claims 17-18 and 20-23

Independent claim 17 requires the elements of means to represent a function in a state flow diagram and means to call the graphical function in a simulation of at least one finite state machine. The Examiner considers that Kodosky discloses the element of means to represent the function in a state flow diagram at page 2, paragraph 16, lines 4-9. However, in the cited section, Kodosky merely discusses that a state diagram having states and transition but does not disclose means to represent a function in a state flow diagram. The Examiner further considers that Kodosky discloses the element of means to use the graphical function from at least one state

machine at page 15, paragraph 166, lines 13-20. However, in the cited section, Kodosky discusses that a state diagram editor may support execution highlighting. While a state diagram editor may support a number of features, none of the ones disclosed in Kodosky bear on being able to call the graphical function from at least one finite state machine. Calling a function from at least one finite state machine is patently distinct from using a feature of the state diagram editor. That is, the state diagram editor is an environment, in which the finite state machine is processed and disclosing that the state diagram editor supports execution highlighting says nothing about what is called from the finite state machine itself.

Accordingly, Kodosky does not disclose each and every element and limitation of independent claim 17. Applicants respectfully request that the Examiner reconsider and withdraw the rejection of independent claim 17.

Applicants note that dependent claims 18 and 20-23 also recite separate patentable subject matter. For example, claim 23 requires the ability to hide the display of the function flow diagram based upon user input. The Examiner considers that Kodosky discloses this element at page 15, paragraph 169, lines 7-12 because the linking of non-graphical code does not involve adding it to the graphical program therefore it is hidden in the graphical environment. The Examiner seems to imply that the non-graphical code is the equivalent of a function flow diagram in the claimed invention. Applicants respectfully submit that non-graphical code is not an equivalent of a function flow diagram, especially considering the limitation that such input define at least one graphical function.

Dependent claims 18 and 20-23 depend on independent claim 17, and are not anticipated by Kodosky for at least the same reasons as above in regard to claim 17. Therefore, Applicants respectfully request that the Examiner reconsider and withdraw the rejection of claim 17 and its dependent claims 18 and 20-23.

Claims 24-31

Independent claim 24 requires the step of using a graphical user interface to create a graphical representation of a finite state machine including a graphical representation of a function. The Examiner considers that Kodosky discloses this step at page 1, paragraph 9, lines

9-14. However, in the cited section, Kodosky only discloses that a graphical program may be constructed using a block diagram editor, and data structures may be automatically constructed to characterize the execution procedure of a state diagram. Applicants respectfully submit that Kodosky does not disclose the possibility of including a function within a state diagram, whereas claim 24 requires not just a function, but a graphical representation of a function within the graphical representation of a finite state machine.

Furthermore, Kodosky does not teach or suggest "calling the function from the executable model of the system during the act of simulating the system represented by the finite state machine." Kodosky does not go into details of simulating the executable system, but it is clear that none of the elements of the system of Kodosky are capable of calling graphical functions from the executable model.

Accordingly, Kodosky does not disclose each and every element and limitation of independent claim 24. Applicants respectfully request that the Examiner reconsider and withdraw the rejection of claim 24 and its dependent claims 25-30. Claim 31 is canceled, hence the rejection is moot.

Claims 32-33

Independent claim 32 requires the limitations of receiving through a graphical user interface a graphical representation of a finite state machine including a graphical representation of a function and calling the function from at least one place in the executable model during the system simulation. As discussed above in connection with claim 24, Kodosky discloses neither one of these limitations. Accordingly, Kodosky does not disclose each and every element and limitation of independent claim 32. Applicants respectfully request that the Examiner reconsider and withdraw the rejection of claim 32 and its dependent claim 33.

Claims 34-42

Independent claim 34 require the element and limitation of "providing a function, said function comprising at least two graphical components and being referenced by at least one of the states or at least one of the transitions to call the function...." As discussed above, Kodosky

discusses, at most, that instructions may be called from a state. However, such instructions are specifically shown to be textual, not graphical, and, therefore, Kodosky does not anticipate the limitation of having a function including at least two graphical components, such function callable from a state or transition.

Furthermore, Kodosky does not teach or suggest a mechanism for grouping the instructions inputted by a user. The advantage of using a function is that the function only needs to be defined once, and one can later "call" the function to perform the same set of instructions without the need to repeat the same set of instructions again at other locations in the state diagram. Applicants respectfully submit that Kodosky does not show that a function can be used such that it can be called in a state diagram.

Therefore, Kodosky does not disclose each and every element and limitation of independent claim 34. Dependent claims 36-42 depend on claim 34 and are not anticipated by Kodosky for at least the same reasons as independent claim 34. Applicants respectfully request that the Examiner reconsider and withdraw the rejection of claims 34 and 36-42.

Claims 43-51

Independent claim 43 recites "a component representing a graphical function and referenced by at least one of the states or at least one of the transitions." As discussed above, Kodosky does not teach or suggest graphical functions or referencing such functions from states or transitions in a graphical model. Therefore, Kodosky does not anticipate claim 43. Claims 44-51 depend on independent claim 43 and are patentable for at least the same reasons. Applicants respectfully request that the Examiner reconsider and withdraw the rejection of claims 43-51.

Claims 52-60

Independent claim 52 recites a limitation of "providing a block component representing a graphical function and referenced by at least one of the states or at least one of the transitions to call the function at one of the states or one of the transitions during execution of the event-driven system." As discussed above, Kodosky does not teach or suggest providing a block component

representing a graphical function. Furthermore, Kodosky is silent on calling the function during execution of the event-driven system.

Therefore, Kodosky does not anticipate independent claim 52. Dependent claims 53-60 are patentable over Kodosky for at least the same reasons as independent claim 52. Applicants respectfully request that the Examiner reconsider and withdraw the rejections of claims 52-60.

Claims 61-78

Independent claims 61, 66, 70 and 74 all recite a graphically defined function, which may be textually called (or referenced) within an executable model. As discussed above, Kodosky does not teach or suggest neither graphically defined functions, not being able to really reference such a function from within a model.

Examiner suggests that Kodosky teaches textually invoking a graphical function at paragraph 132, lines 1-5 and Figure 8. Applicants respectfully disagree. The cited section of Kodosky indicate that a user may "specify exactly what set of instructions should be executed when this state becomes active." Neither the cited text, nor Figure 8 teach or suggest that any of the specified instructions may refer to a graphically defined function. Figure 8 seems to teach away from having graphical functions, by specifically illustrating text input fields for any of the instructions that a user may want to enter. While some of those instructions may be function calls, the function called are textual functions, not graphical functions, as required by independent claims 61, 66, 70 and 74. No section of Kodosky teaches or suggests the ability to textually call a graphically defined function from within an executable model.

Therefore, Kodosky does not anticipate independent claims 61, 66, 70 and 74. Dependent claims 62-65, 67-69, 71-73 and 75-78 depend on independent claims 61, 66, 70 and 74, correspondingly, and, therefore, are patentable over Kodosky for at least the same reasons as above. Applicants respectfully request that the Examiner reconsider and withdraw rejections of claims 61-78.

CONCLUSION

In view of the above amendment, Applicants believe the pending application is in condition for allowance and urges the Examiner to pass the claims to allowance. Should the Examiner feel that a teleconference would expedite the prosecution of this application, the Examiner is urged to contact the Applicants' attorney at (617) 227-7400.

Please charge any shortage or credit any overpayment of fees to our Deposit Account No. 12-0080 under Order No. MWS-070RCE. In the event that a petition for an extension of time is required to be submitted herewith, and the requisite petition does not accompany this response, the undersigned thereby petitions under 37 C.F.R. §1.136(a) for an extension of time for as many months as are required to render this submission timely. Any fee due is authorized to be charged to the aforementioned Deposit Account.

Dated: **May 2, 2007**

Respectfully submitted,

By 

EuiHoon Lee
Registration No. L0248 for
Kevin J. Canning
Registration No. 35,470
LAHIVE & COCKFIELD, LLP
One Post Office Square
Boston, Massachusetts 02109
(617) 227-7400
(617) 742-4214 (Fax)
Attorneys for Applicant